# Visualizing Dynamic Networks with Matrix Cubes

Benjamin Bach, Emmanuel Pietriga, Jean-Daniel Fekete

# Visualizing Dynamic Networks with Matrix Cubes

**Benjamin Bach**
INRIA
benjamin.bach@inria.fr

**Emmanuel Pietriga**
INRIA & INRIA Chile (CIRIC)
emmanuel.pietriga@inria.fr

**Jean-Daniel Fekete**
INRIA
jean-daniel.fekete@inria.fr

## ABSTRACT

Designing visualizations of dynamic networks is challenging, both because the data sets tend to be complex and because the tasks associated with them are often cognitively demanding. We introduce the *Matrix Cube*, a novel visual representation and navigation model for dynamic networks, inspired by the way people comprehend and manipulate physical cubes. Users can change their perspective on the data by rotating or decomposing the 3D cube. These manipulations can produce a range of different 2D visualizations that emphasize specific aspects of the dynamic network suited to particular analysis tasks. We describe Matrix Cubes and the interactions that can be performed on them in the *Cubix* system. We then show how two domain experts, an astronomer and a neurologist, used Cubix to explore and report on their own network data.

### Author Keywords

Dynamic Networks; Information Visualization; Interaction; Metaphors; Multiple Views

### ACM Classification Keywords

H.5.2 [Information Interfaces and Presentation]: User Interfaces - Graphical user interfaces.

### General Terms

Human Factors; Design

## INTRODUCTION

Network structures exist in many domains, including social networks, neural signaling networks in the brain, and ad-hoc networks of computers and mobile devices. To understand and analyse these networks, it is important to be able to visualize them in a comprehensible way. Yet, designing network visualizations that effectively support users in their exploration and analysis tasks is quite challenging. Moreover, many networks are actually dynamic, and their topology as well as the attributes associated with nodes and edges can vary over time. This additional level of complexity often leads to more complicated visual interfaces with larger numbers of views and more widgets to adjust parameters of the visualizations. This is a significant issue, as above a certain level of complexity, users are less able to learn – or less willing to take the time to learn – the system.


*CHI 2014*, April 26 - May 01 2014, Toronto, ON, Canada

http://dx.doi.org/10.1145/2556288.2557010

One of the main goals of human-computer interaction is to improve the tradeoff between power and simplicity in interfaces [3]. In this paper we introduce the *Matrix Cube*, an easy-to-understand yet powerful approach to representing dynamic networks. Matrix Cubes result from stacking adjacency matrix representations of the network at each time step, forming a space time cube. Compared to node-link diagrams, adjacency matrices are better suited to the visualization of dense networks. With the Matrix Cube, we explore their applicability to the visualization of dynamic networks.

The Matrix Cube representation and associated interactions are inspired by the way people comprehend and manipulate physical cubes. Users can change their perspective on the data by rotating and projecting the 3D cube, or decomposing it using operations such as slicing and filtering. These operations lead to various 2D visualizations, that offer a set of coherent, easy-to-relate views on the data. The 3D view of the cube provides an overview of the data, as well as a visual model for understanding transitions and pivots between views. Meanwhile, the 2D views derived from the cube support more detailed analysis of specific aspects of the network and entities that constitute it.

In this paper, we first review related work on visualizing dynamic networks and introduce the concept of Matrix Cubes. When then present *Cubix*, an interactive system that uses Matrix Cubes to support the visualization and analysis of large, dense dynamic networks. Finally, we describe how two domain experts—an astronomer and a neurologist—used Cubix to analyze dense dynamic network data from their own work.

## BACKGROUND

We define a dynamic network as $G = (V, E, T)$ where $V$ is the set of all vertices, $E$ the set of all edges, and $T$ the set of discrete time steps. Edges can have attributes associated with them, such as a weight. Attributes can vary between time steps. A time step $G_t = (V_t, E_t)$ of $G$ consists of the vertices and edges present in the network at time step $t \in T$.

### Node-link diagrams

Most techniques for exploring dynamic networks visualize them using node-link diagrams. Any static network tool can be used to create an image per time step. These diagrams can then be displayed one after the other (time-multiplexed), or side by side (juxtaposed). Animations between individual images, as in Ahn *et al.* [1] and Marey [16], can make it easier to recognize changes and better track moving nodes and edges when the graph layout is not stable. But showing images in a sequence can increase users' cognitive load by forcing them to remember the network's state when navigating

between time steps. This is especially problematic for networks with many time steps. Alternatively, individual images can be shown side-by-side as in Ploceus [25]. The amount of screen real-estate dedicated to each time step depends on their number. This has a direct impact on the amount of detail that can be displayed. While these side-by-side images provide a quick overview of the network's evolution, they make it hard to track individual graph elements across steps, or directly compare distant images, especially for dense graphs.

Changes in time-multiplexed or juxtaposed representations are not visualized explicitly, which makes them difficult to identify. Events such as the addition or removal of a node or edge can be explicitly encoded in a single image using color, as in Gevol [11] or Donatien [21]. However, these visual variables are no longer available to encode information about the network, such as edge weight [2].

Dynamic networks can be decomposed not only along time, but also based on their topology. *Dynamic ego-networks* representations show the evolution of a node's neighborhood using other approaches (e.g. a node-link diagram that maps time to concentric circles as in Farrugia *et al.* [14], or a timeline as in Shi *et al.* [29]). Ego-networks make it easier to observe specific nodes, but do not help identify and understand the more general evolution of trends over the network. Other techniques represent the entire network along a single timeline using a spatial mapping that makes it possible to describe locations and distances between time points. Using this approach, the challenge is to represent the topology using the only remaining spatial dimension. Reda *et al.* [27], as well as Sallaberry *et al.* [28], represent bundles of clusters or communities along a timeline. Parallel edge splatting [7] shows time steps as vertical lines and places network links between slices. Such timeline visualizations can show temporal trends, but fail to show the network's topology.

#### Adjacency Matrices

Adjacency matrices provide an alternative to node-link diagrams that is particularly effective when visualizing dense graphs [2, 18, 22]. Furthermore, matrix cells provide better opportunities than the lines or curves of a node-link diagram to visually encode information about the corresponding edges. However, only a few projects have investigated the use of matrices to visualize dynamic networks. TimeMatrix [31] displays a temporal bar chart in each matrix cell to show the evolution of edge weights. Brandes and Nick [6] use *gestalt-lines*, to represent changes of bi-directed edge weights over time. Another way to encode edge value changes over time is to use brightness changes on space-filling curves in matrix cells [30]. In addition to matrices, Netvisia [20] uses a heatmap, where one dimension is time and the other contains the vertices. Similar to FlowStrates [5], cells in the heatmap encode the value of one particular vertex attribute in the corresponding time step.

Each of these techniques emphasize some aspects of dynamic networks at the expense of others. Common tradeoffs include visualizing the network's topology vs. encoding time, providing overview vs. detail, and showing many time steps with few details vs. fewer ones with more detail.

#### The Space-Time-Cube Metaphor

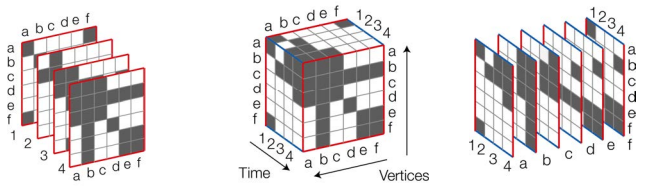
Metaphors are used frequently in interface design to help users understand a new or unknown system or object by referring to existing systems with which they are already familiar. This provides a mental model of the system's functionality that helps users to understand the *current state* of the system and to *anticipate interactions* that might change its state. For example, both ScatterDice [13] and GraphDice [4] use a rotating cube metaphor to convey changing dimensions in scatterplots and multivariate graphs, respectively.

A space-time cube is a representation that maps data onto two dimensions, while time is shown as a third spatial dimension. Space-time cubes are able to show both spatial *and* temporal information simultaneously [17, 23, 24]. However, those representations face several common problems encountered when showing 3D models on a 2D screen: more complex navigation, reduced depth perception, and visual occlusion. To address the latter problem, GeoTime [23] provides an additional view that shows trajectories as projections on the landscape. While space-time cubes in geographic information systems are often sparse, time cubes for video analysis, resulting from the alignment of all video frames, are fully filled. To explore their interior, Fels et al. [15] allow users to freely cut the video cube. Cassinelli et al. [10] interactively distort the cube. Carpendale investigated several ways to explore dense time cubes using geometric distortion [9], cutting planes, and filtering techniques [8].

Brandes and Corman [6], as well as Dwyer and Gallagher [12], represent dynamic networks as space-time cubes by extruding nodes from 2D node-link diagrams to 3D columns and "worms". Edges become bridges between extruded nodes. However, node-link diagrams become unreadable above a certain (very low) network density threshold and stacking them in 3D introduces even more line crossings due to perspective overlap. Interactive 3D rotation or temporal coloring fail to fully address these issues. In addition, projection and decomposition of these diagrams is not straightforward, which makes visual exploration and analysis difficult.

## THE MATRIX CUBE

The *Matrix Cube* is a representation of dynamic networks based on the space-time cube metaphor. It is created by stacking adjacency matrices in chronological order, one for each time step (Figure 1-a and 1-b). The cube consequently has two vertex dimensions which are colored red, and one time dimension which is colored blue. A *cell* $c_{vwt} \in (V \times V \times T)$ exists for each edge $e_{vwt}$ between vertices $v$ and $w$ at time $t$. Along the time dimension, the cube contains *time slices* (adjacency matrices) that show the time steps of $G$ at time t (Figure 1-a). Slicing the cube along one of its vertex dimensions yields *vertex slices*, one for every vertex (Figure 1-c). Rows in a vertex slice correspond to the network's vertices $V$ and columns correspond to time steps in $T$. Vertex slices show the dynamic ego-network of each node; a cell in the vertex slice for vertex $v$ indicates an edge between $v$ and the vertex indicated by the row the cell belongs to, at the time indicated by the cell's column. Since the Matrix Cube comprises two vertex dimensions, there are potentially two different types of




(a) Adjacency Matrices (b) Matrix Cube (c) Vertex Slices

**Figure 1. The Matrix Cube. (a) Each time step of the network (1,2,3,4), is represented as an adjacency matrix. (b) The Matrix Cube results from stacking those matrices. Red edges of the cube hold nodes and correspond to the rows and columns of the constituent adjacency matrices; blue edges of the cube hold time steps. (c) Slicing the cube along one of the vertex dimensions yields *vertex slices*.**

vertex slices. We focus on undirected networks, where both types of vertex slices contain the same information.

Cells in the same row and column across all matrices form a *time vector* that describes the evolution of the connectivity between two vertices. In a similar way, cells in the same row or column in one time slice form a *neighborhood vector* that describes the neighborhood of a single vertex at the time corresponding to this slice.

To make topological patterns visible in adjacency matrices as well as in matrix cubes, rows and columns (vertex slice) can be reordered using one of four methods: alphabetical, reverse Cuthill-McKee, TSP, and optimal leaf-ordering of a hierarchical clustering technique.

## CUBIX

Cubix is an interactive system for the visualization of large, dense dynamic networks represented using matrix cubes. It lets users navigate in the data by manipulating the cube interactively, thus deriving meaningful 2D views that emphasize particular dimensions of the data. Orientation in, and navigation between, views is facilitated by the *Cubelet*, a small widget that represents the current view configuration and all possible manipulations. Cubix animates the 3D representation of the cube to smoothly pivot between views, illustrating the operations performed during the transitions and helping users maintain perceptual continuity across multiple views.

Cubix is based on the following design principles:

1. **3D as a pivot** — As discussed earlier, 3D representations are hard to visualize on two-dimensional screens. Cubix focuses on the 2D visualizations that can be derived from the cube. Still, explicitly showing the 3D cube has advantages: 1) it explicitly conveys the cube-based model to the user, 2) it provides an overview of the data along all dimensions, and 3) it serves as a pivot visualization when switching between views.
2. **Animated transitions** — Changes between views on the cube are smoothly animated, providing a good level of perceptual continuity during transitions, which helps users to understand the relationship between views as well as track elements across them.
3. **Limited number of views** — The number of views that can be derived by decomposing the cube and arranging its parts is potentially infinite. In Cubix, we constrain the set of operations that users can perform on the cube, in order to keep navigation simple and manageable.
4. **Easy navigation** — Some views require multiple operations. For instance, laying out slices side-by-side requires specifying the slicing direction (time slices or vertex slices), and moving them to their appropriate position in 3D space. The cubelet allows users to quickly navigate between most predefined views, which are also accessible via keyboard shortcuts.

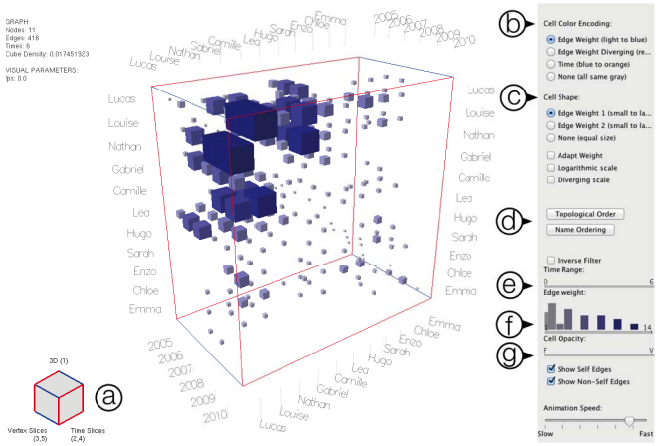



**Figure 2. Cubix UI screenshot. a) Cubelet Widget, b) Cell color encoding, c) Cell shape encoding, d) Vertex ordering, e) Time range slider, f) Cell weight filter with histogram indicating edge weight distribution, g) Cell opacity.**

We illustrate Cubix using a simple research collaboration network. This network contains one time step per year. Researchers correspond to vertices, and connections between vertices indicate a co-authorship relation on at least one publication during that year. Self-edges correspond to publications by a single author. Edge weights represent the number of papers between any two authors for a given year.

### Matrix Cube Overview

Figure 2 shows the default view for this co-authorship network. The main interface components include: the Matrix Cube in the center, the Cubelet widget for rapid view switching (a), and control widgets on the right-hand side to parameterize the visualization (b-g).

For this dataset, the cube's axes are labeled with author names and dates. By default, cell size and color indicate the number of publications per year (edge weight). Color ranges from light gray for low edge weight, to dark blue for high edge weight (*value encoding*). Edges with high values stand out across the entire cube while smaller cells make it possible to see through the cube. While freely rotating the cube can give an overview of the data, it is often difficult to locate particular cells of interest, and to situate them along the time dimension. To tackle this problem, we can redundantly encode time by switching to a cell color mapping (*time encoding*, Figure 2-b) that uses a gradient ranging from blue (early time steps) via violet to orange (later time steps).

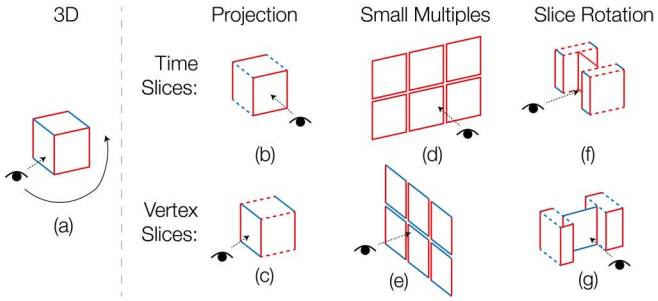



**Figure 3. Cubix View design space. Columns indicate operations applied to the cube. Rows indicate operations applied to time (red × red) and vertex slices (blue × red), respectively. (a) 3D view, (b) time-projection view, (c) vertex-projection view, (d) time small multiples, (e) vertex small multiples, (f) time-slice-rotation, and (g) vertex-slice-rotation.**

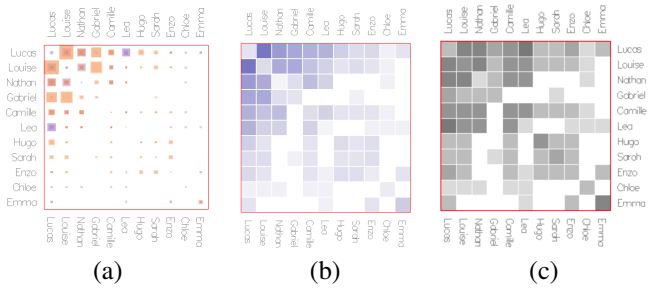



**Figure 4. Cell color and size mappings in Time-projection view. (a) Cell color is mapped to time, and cell size to edge weight. (b) Constant cell size, with color mapped to edge weight shows accumulated edge weight over time. (c) Constant cell size and same color (gray) for all cells gives an idea of the number of edges in time vectors, independent of weight.**

By looking at the cube in 3D, one can get an overview of the data, including the number of vertices in the network (rows and columns), the number of time steps, the number and density of edges (cells), and the distribution of edges over time. Rotating the cube gives a better impression of particular cells' location, and helps identify sparse and dense areas. From Figure 2 we can notice a spatio-temporal cluster that features a dense core of high-weight edges and a wide range of somewhat lower-weight edges. A significant number of smaller cells exist, almost equally distributed across the cube.

While the 3D view shows the entire dataset, it only shows the most obvious patterns. Other views can be derived from this default representation, which emphasize specific aspects of the network. Cubix currently implements seven predefined views, illustrated in Figure 3. Slice and camera positions are predefined, as well as the projection method (perspective projection for the 3D view, orthogonal projection in all other cases). Each view provides specific interaction capabilities, such as slice rotation or free camera rotation.

### View Navigation with the Cubelet Widget

Quick navigation between views is enabled by an interactive overview widget called the *Cubelet* (Figure 2-a). The Cubelet is a static 2D-isometric abstraction of the Matrix Cube that indicates the current view of the Matrix Cube. It uses the same color coding for the cube's dimensions as the actual Matrix Cube: blue for time, red for nodes. The Cubelet's left face corresponds to vertex slices (red × blue), the right face to time slices (red × red). Clicking on one of these faces transitions to the corresponding projection view. During the transition, the Matrix Cube smoothly rotates to show the selected side. In any of the projection views, the corresponding face of the Cubelet is highlighted. Dragging the mouse on the cube's faces transitions to the corresponding small-multiples view. When this happens, the Cubelet also changes its appearance, to make it look like if it were sliced. Clicking on the Cubelet's top face animates back to the 3D pivot.

### Exploring the Network's Aggregated Topology

Clicking on the Cubelet's right face rotates the cube to the time-projection view shown in Figure 4-a. Cubix switches from a perspective projection to an orthogonal projection. Consequently, all cells corresponding to the same edge over time become aligned (all cells in the same time vector become superimposed), and the topological structure of the network becomes clearly visible. The opacity slider (Figure 2-g) can be used to adjust cell translucency. Using the time color mapping, cells that are predominantly purple indicate an edge weight that decreases over time, while those that are predominantly orange indicate an increasing one.

Superimposed cells of different sizes appear nested inside of one another. A higher number of nested squares indicates more weight variability. Figure 4-c demonstrates an alternative cell color-and-size mapping that makes use of translucency, enabling users to evaluate how often an edge is present between two nodes over time. Darker cells indicate a longer presence or higher overall weight, which in this case translates to a larger number of co-publications between these two authors.

### Temporal Trends

Clicking on the Cubelet's left side rotates the Matrix Cube by 90 degrees to the right (Figure 5). In this view, we can see a heat-map with one column per time step (left to right). Rows still represent the network's vertices, showing the aggregated connectivity for each vertex (neighborhood vector), for every year. As illustrated in Figure 5, the general trend in this research group is towards more publications and collaboration (darker cells due to superimposition of translucent elements), except in 2008. We can further compare the connectivity over time for each vertex by comparing rows in this view. We can see that some authors always collaborated and did so increasingly over time (Lucas, Louise, Nathan), while others joined the team later (Gabriel, Hugo, Sarah, Enzo), or collaborated less frequently (Chloe).

### Topology of Individual Time steps

We can use the time range slider to change the set of time slices that are visible (Figure 2-e). A list next to the cube in the time-projection view shows the labels of elements in the projected dimension (Figure 5-b). Hovering over a label in this list shows only the corresponding slice.

This allows for quick navigation to, and between, time slices. If we want to see the different stages of topology evolution at-a-glance, dragging the mouse on the Cubelet's right face

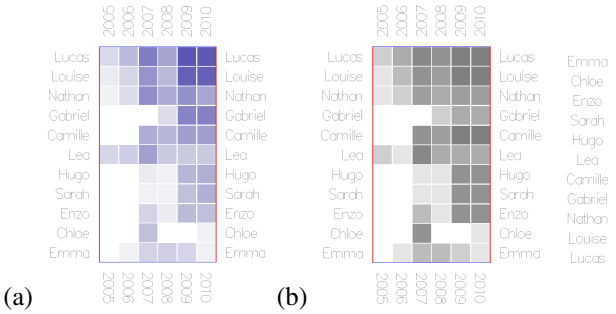



**Figure 5. Temporal trends in the vertex-projection view**

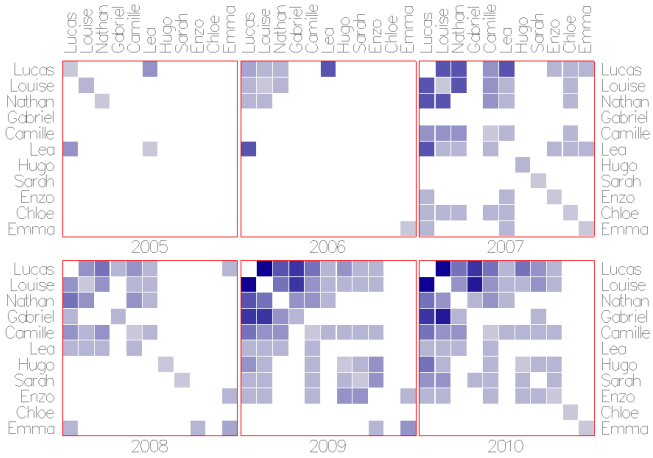



**Figure 6. Time slices juxtaposed in the time-slices view. Darker cells correspond to more publications between people.**

triggers an animation that decomposes the cube into individual time slices. Switching back to the edge-weight-to-color visual mapping, Figure 6 reveals four characteristic topological configurations, that explain the different patterns observed in Figure 5. Year 2007 seems to contain meaningful patterns, that the default reordering of rows and columns, computed globally over all time slices, fails to show (Figure 6). When only a subset of time steps are selected, asking for a new reordering takes only those steps into account, potentially yielding more meaningful visual patterns.

### Slice Rotation

Brushing through slices using the list displayed on the right-hand side in both projection views makes it possible to quickly browse through them one-at-a-time. However, all information about other slices is hidden. Projected slices (orthogonal to the camera) can be rotated 90 degrees, as illustrated in Figure 7 (cf. Figure 3-b and Figure 3-c), thus creating a focus+context view. Users can rotate as many slices as they want simultaneously. In a rotated slice, columns correspond to those of the projected dimension. Rows are still horizontally aligned with those of the projected slices. For example, the rotated slice in Figure 7 reveals that Camille collaborated with three people from the core cluster (Lucas, Louise, Nathan) over four consecutive years (2007-2010).

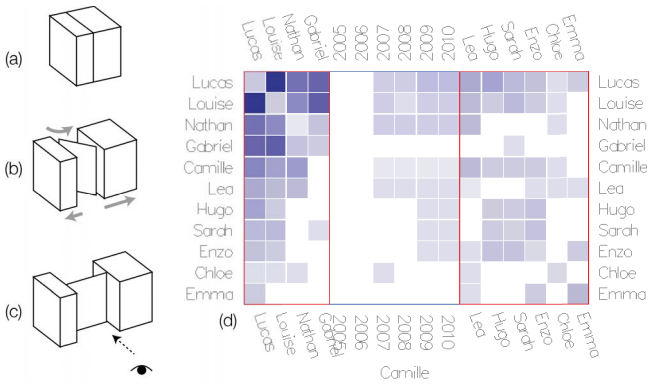



**Figure 7. Rotation of a vertex slice. (a-c) schematic representation of the operation, (d) resulting focus+context visualization.**

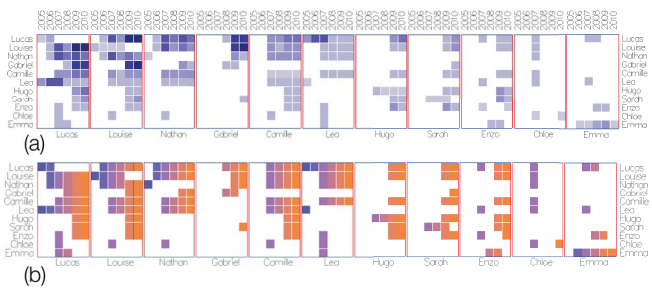



**Figure 8. Vertex Small multiples view illustrating both color mappings. (a) Cells are colored according to edge weight, highlighting differences in weight across slices. (b) Mapping color to time-index facilitates topological comparisons across slices.**

### Individual Collaboration Behavior

Switching to the vertex small-multiples view lets us compare all collaboration patterns across group members (Figure 8-a). The overview on all vertex slices reveals who has been most active and for what period of time. Slices for Hugo, Sarah and Enzo show very similar patterns. The same is true for Lucas, Louise and Camille. Switching to the time-index color mapping (Figure 8-b) makes it easier to compare ego-network topologies for a given time step.

### Cell Filtering

Cubix offers multiple filtering methods that let users specify what cells should be shown or hidden in the cube (and thus across all views). (i) Selecting one or more slices, by right clicking slice labels in the 3D view, automatically hides all others. If both horizontal and vertical slices are selected, only the cells at the intersection of all selected slices remain visible. Hovering over a hidden slice's label temporarily reveals that slice. (ii) Clicking a cell once hides all slices but those that contain this cell. Clicking it twice hides all cells but those that belong to the same time and neighborhood vectors. (iii) The edge weight range slider makes it possible to hide edges whose weight is below or above some thresholds, which is especially useful when the cube is complete. Cells filtered out using one of these methods can be made translucent instead of completely invisible, so that some context is retained. Finally, individual slices can be painted with a custom color, making them easier to keep track of across views.

## VALIDATION WITH EXPERTS

Visualizing dense, weighted dynamic networks is a problem in multiple application domains. For this first evaluation of matrix cubes, we contacted two researchers that face the issue of visualizing dynamic networks: an astronomer and a neurologist. We wanted to focus on the overall usefulness of Matrix Cubes and usability of Cubix through an open-ended evaluation using real data, so as to answer questions such as: Is the metaphor understandable? How well do experts manage to manipulate the cube and interpret the views? Can they gain insights about their data from this visualization?

### General Methodology

Both experts were approached independently and asked if they would be interested in trying to visualize their data with Cubix. We demonstrated Cubix using the co-authorship network described earlier in this paper. In both cases, the experts' feeling was that the approach had potential, and they agreed to provide us with actual data. Once their data had been imported in Cubix, we met again with each of the two experts for a longer session, walking them through the different views and features based on a sample of their own data. Experts then took control, looking at their data in more detail by themselves under our guidance as detailed below.

### ALMA

The Atacama Large Millimeter/submillimeter Array [26] is a state-of-the-art radio-telescope composed of 66 high-precision *antennas* currently under construction in the Chilean Andes. Observations are based on the principle of interferometry: a source in the sky is observed by a subset of 12-to-50 of those antennas, called an *array*. In a given array, all antennas are connected to the other ones in a pairwise manner, thus forming a fully-connected network. Edges of this network, called *baselines*, have several quantitative time-dependent variables associated with them, which astronomers are interested in looking at.

Astronomers currently use a tool developed in-house to plot these variables as scatterplots or line charts. While these charts may be able to produce legible visualizations for antenna-based variables, they do not scale to the number of baselines in an array (typically 1225 for 50 antennas). Still, astronomers need to either monitor in real-time or analyze a posteriori those data over numerous time steps.

Following the general methodology described above, we approached an ALMA astronomer. The session consisted of two parts. We used one of the four baseline-based variables available in the data he had provided (see below) for the walkthrough. Then the astronomer took control of the computer, and visualized all four datasets, commenting on his findings and insights, and asking for help to get to a particular view when stuck (he was told that he could freely ask for our assistance). These two phases lasted 22 and 45 minutes, respectively. The entire session lasted 1.5 hours, including preliminary explanations and post-hoc discussions. We recorded the session and took pictures of meaningful views on the data for analysis once back in our lab.

#### *Data*

The dataset corresponds to a calibration, used to compute very precisely the (x,y,z) geographical position of antennas. For this, antennas point at a series of quasars that act as reference sources. Quasars are pointed at, one after another, by all antennas simultaneously. Each quasar represents one of the 34 time steps in the data. The data consists of four variables, each visualized independently in a separate cube: DELAY, Amplitude (AMP), Amplitude RMS (RMS), and signal-to-noise ratio (SNR). All are measured for 42 antennas, i.e., 861 baselines, for one particular baseband and one polarization only. Each cube thus consists of $42 \times 42 \times 34 \approx 60{,}000$ cells (see Figure 9-a).

#### *Usage Scenario*

We first provide a summary of the visual exploration process over the entire session, focusing on the main steps related to the key findings which are discussed afterwards.

We used the RMS data to give a walkthrough to the astronomer. Straight from the 3D overview, four time slices stood out, featuring larger values than other time steps, as illustrated in Figure 9-a. After briefly showing the time-projection view, we quickly switched to the vertex-projection view at the expert's request, as he was interested in getting a better look at the evolution of values over time (Figure 9-b). From this view, he confirmed his intuition that the four time slices spotted in the 3D overview were quasars that were too weak and should probably be removed from the catalogue of sources used for calibration. Based on the visual appearance of the four columns in the vertex-projection view, he further inferred that these four time steps might actually correspond to the same quasar observed 4 times, or to 2 quasars observed twice each. At the expert's request, we filtered out higher values corresponding to the weak sources, so as to be able to better compare the cells from other time steps visually (the value-to-color mapping adjusts cell color dynamically as higher-value cells get filtered out). The result looked homogeneous, leading the astronomer to the conclusion that there was no antenna-based problem, which would have appeared visually as one-or-more entire rows of cells with high values. We then finished the walkthrough by explaining both time and vertex small-multiples views, illustrating features such as brushing.

Now familiar with Cubix's main features, the astronomer started exploring the data on his own, switching to DELAY. In this data, he was looking for baselines whose delay varies over time, as this was not supposed to happen (delay should be more or less constant). He switched to the vertex-projection view, which shows time steps as columns. However, as all vertex slices were superimposed, it was difficult to spot specific baselines that would feature significant variability. The expert realized this, and decided to switch to the time-projection view. Some rows and columns stood out, indicating possible antenna-based problems. From there, he asked that we switch to the view that shows vertex slices as small multiples, as he wanted to get an overview of what particular antennas feature higher variability, something that can easily be seen in this view by visually scanning the whole

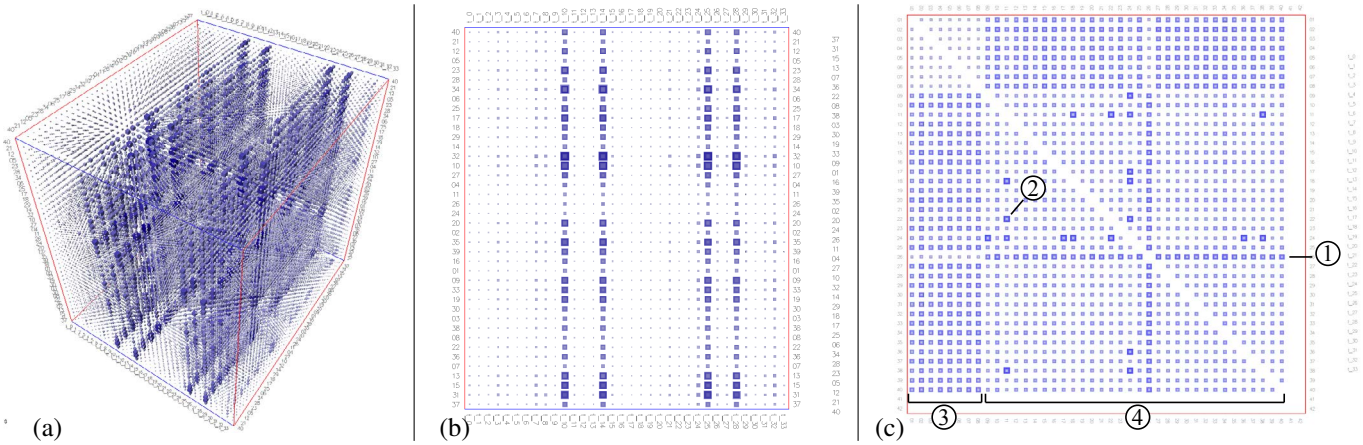



**Figure 9. ALMA dataset. (a) Overview of RMS in which four time steps seem to stand out. (b) The vertex-projection view of RMS confirms this and provides additional detail. (c) The time-projection view on AMP, showing an abnormal behavior of one specific antenna (1), various baseline specific outliers such as (2). A different behavior of the first eight antennas compared to the other ones is clearly visible: (3) vs. (4).**

set of small multiples (Figure 10). This view revealed another pattern: for some antennas, roughly half of the baselines had much larger delays than the other half. The astronomer switched back to stacked vertex slices to quickly brush through the slices in more detail. There was no clear explanation for this observation, which requires further investigation by looking at additional data not available here. We then switched to AMP data. The astronomer was very interested in the time-projection view, as the two main types of problems were readily apparent. First, antenna-based problems affect all baselines that connect to it, which visually translates into the corresponding row/column featuring a lot of variability (see (1)). Second, problems affecting a single baseline, possibly due to, e.g., atmospherical changes, which visually translate into isolated cells featuring a lot of variability (see (2)). Another pattern that clearly stood out in this view is the difference between the eight left-most columns (3) and the following ones (4), attributed to the different types of antennas that compose ALMA[1]. Finally, the astronomer looked at SNR data, identifying similar visual patterns, and then finished the session.

### Brain Function Activity

The second expert, a researcher in neurology, studies signal patterns between regions of the brain under different stimuli, collected using EEG-correlated fMRI [19]. He is interested in identifying time steps that feature a high activity level overall, connectivity patterns that reoccur over time, and which regions are active at specific moments. All of these can then be compared across different stimuli.

The neurologist currently uses spatial brain maps to explore his data, but the physical position of brain regions is not important when analyzing this type of dataset: maps do not properly convey the network's topology. Currently, no visualization faithfully represents his data since node-link diagrams fail to provide a legible view of such dense networks.

[1]We discussed the possibility that the antenna in (1) could belong to group (3). This was ruled out because of the way the data was generated, with antennas sorted alphabetically, and thus by type.

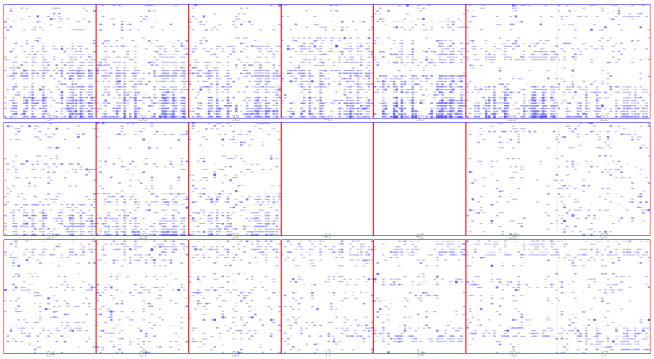


**Figure 10. ALMA DELAY data: vertex slices as small multiples. View cropped to show only 21 out of 42 slices. Similar patterns (reversed) appear in the lower half.**

Therefore, just like the astronomy expert, neurologists typically look at time-series for each region of the brain, without any visual feedback about the topology.

The methodology employed for the neurologist was slightly different than for the first expert. The neurologist was located on another continent, and thus sessions were conducted remotely using a videoconferencing tool. The neurologist was provided with the Cubix software; the system ran simultaneously on both ends, with screen sharing enabled. We conducted two sessions, with one day in-between the two. After looking at the data and explaining to us what it was about, the neurologist asked for a few changes to the visual representation to reflect domain-specific information. We planned for the second session to implement the requested changes and give him time to explore his data with the adapted version of Cubix. The second session was conducted under the same conditions, using the new visual representation.

#### *Data*

We obtained data about three different regions of the brain from multiple subjects, observed under 3 different stimuli. For each of these regions, signals for 50 different frequencies in 8-to-16 subregions were recorded. Several measures are used to characterize connectivity, such as Granger causal-

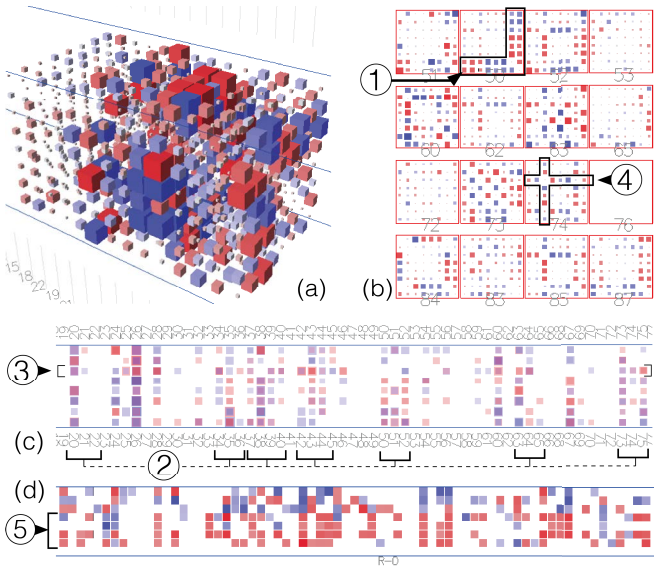



**Figure 11. Brain connectivity data. 8 regions over a sample of 15 time steps. (a) Overview reveals that some time steps are more active than others. (b) Detailed view of 16 time slices. (c) Vertex-projection view, showing activity patterns and revealing one region more active than the others. (d) Individual vertex slice sending more than it receives.**

ity, and we can only show one dynamic network at a time from this rich dataset in a matrix cube. The networks visualized in Cubix consist of 8 or 16 vertices over 150 time steps (5 minutes of recorded brain activity at a 2-second sampling frequency). Measures exist between any two regions, meaning that networks are fully-connected, the corresponding matrix cubes containing from $8 \times 8 \times 150 = 9600$ to $16 \times 16 \times 150 = 38400$ cells. Figure 11-a shows 15 out of 150 time steps for a network of 8 regions.

During the first session, the neurologist explained that edge weights have normalized values in $[-1.0, 1.0]$, and that edges of interest are those near the extrema. We added an additional color scale and mapping, emphasizing those values, as illustrated in Figure 11: 1.0 gets mapped to red, 0 to light gray, and $-1.0$ to blue. A new mapping of edge weight to cell size was also added, assigning smaller sizes to edges with lower absolute values. The range slider used to filter out edges based on their weight now features an option to invert the selection, so as to remove values in-between the two knobs rather than those outside the corresponding range.

### *Usage Scenario*

Data visualized during the first session corresponded to a resting phase (no visual stimulus, subjects' eyes closed). At the neurologist's request, the second session focused on data observed during a phase featuring visual stimuli. Using the new color and size mappings revealed the symmetric distribution of red and blue cells in the 3D overview (Figure 11-a). Several time steps also stood out as significantly denser than others, distributed over time. Switching to time-projection view made it possible to get a clear look at the denser time-slices, confirming the symmetry between red (sending) and blue (receiving) edges. Beyond symmetry, one pattern appeared consistently across all slices that feature a high level of activity, made even more apparent when reordering rows and columns based on their similarity: two or three regions received from exactly four other regions (blue cells in the same row) while sending signals to the remaining four regions (red cells in the same row, ①). Sending and receiving regions were the same. However, one of the two active regions also sent to the other one, not receiving any signal from it.

Switching to the vertex-projection view (Figure 11-c), salient patterns could be observed when looking at the temporal distance between highly-active time steps. The neurologist indicated again that edges with low absolute values were irrelevant; those got removed accordingly, keeping cells whose absolute value was larger than $|0.5|$ only. We adapted translucency as illustrated in Figure 11-b. Rotating some time slices in-place enabled the neurologist to recognize the topological patterns previously spotted in the time-projection view. Temporal patterns seemed randomly distributed but occurred in intervals of 3 or 4 time steps ②. As illustrated in Figure 11-c, one region was more active than the others ③. Switching to time small multiples (Figure 11-b) confirmed that region R-4 was indeed active in many time slices, and was often among the most active ones ④. Switching to the vertex small multiples, the neurologist found two regions that were more active than the others, one of them ⑤ sending more often than it received, and almost always sending to the same regions (Figure 11-d).

### Key Findings

**The cube metaphor was quickly understood. It helped, along with animated transitions, to switch between views.** Right from the start, both experts understood what the cube was about, as well as the views that were derived from it. For instance, as the view was animating from the 3D cube to the time-projection view, the astronomer made the following comment: *"Ah! ok, now you are stacking time."* Once he took over, he always managed to switch between views without assistance, sometimes hitting the wrong key but quickly recognizing that this was not the view he wanted. The neurologist used both the Cubelet and keyboard shortcuts. There was only a bit of confusion for the astronomer the first time we switched to the view that shows vertex slices as small multiples, but that view was then well understood and used, though the astronomer asked for assistance to go back to this particular view from the time-projection view. It is worth noting that he knew exactly what view he wanted; his problem was that he did not remember how to get there. Further evidence of the experts' understanding of the matrix cube comes from a comment the astronomer made as he was looking at DELAY data in the time-projection view. As he was looking for baselines featuring significant variability, i.e., squares within squares in a given cell, he realized on his own that for a given baseline, larger cells earlier-in-time would hide smaller cells that were behind them in this particular view, thus masking some types of variability patterns. This is indeed true; we then explained how to overcome this limitation in Cubix by, e.g, adjusting cell translucency to see through them, or by switching to another view. The astronomer also took for granted that cells filtered out in one view would still be hidden when transform-

ing the cube to another view, without us mentioning this at all. We interpret this as further evidence that the cube was well-understood as a model/pivot view.

**Filtering, brushing and linking were useful.** To look at individual slices, the astronomer used the slice list on the right side in vertex- and time-projection views. Sometimes he looked at one particular antenna he had identified in another view, sometimes he browsed through slices to find antennas with particular characteristics hinted at by the default view, which shows all slices stacked. Filtering was also found useful, and the astronomer understood on his own that filtering was happening on the cube as an object, not on a particular view of it. He made one request: in addition to being able to select what slices to show, he wanted to select which slices to hide (which was at the time only possible by selecting all slices not to hide). The astronomer also mentioned that brushing and linking in the small multiple views was very helpful to compare particular values across slices.

**Cell color was always mapped to edge value.** The neurologist asked specifically for a new edge-weight-to-color mapping and used it exclusively. The astronomer was told about both original color mappings: time and value. He immediately stated that redundantly encoding time with color was of no interest to him, instead, he wanted cell color to reflect the associated edge value. The fact that most of the walkthrough was performed with cell size set to its maximum value for all edges (i.e., not reflecting the associated value) might have played a role in this, as there was then no other way in this configuration to visualize edge value.

**Row and column reordering are useful, but can be confusing.** Rows and columns of matrices can be either sorted alphabetically or reordered based on their similarity, so as to make some visual patterns such as clusters more apparent. By default, users are presented with the latter option. The astronomer had not realized the type of ordering, which was confusing to him at first, as he is used to looking at matrices sorted alphabetically, by antenna type and name. As a result, he was not expecting to see some of the visual patterns that appeared, and was instead expecting individual, isolated rows and columns to stand out. He realized this when we switched to *name ordering* while discussing possible differences due to antenna type. This alphabetically-sorted view felt much more familiar to him, as the visual patterns of clusters were no longer there to *"prevent him from seeing individual rows and columns."* However, the neurologist specifically requested that rows be reordered based on similarity when looking for patterns in the time-projection view.

**Matrix Cubes succeeded in providing a legible visualization where other approaches had failed so far.** Astronomers currently visualize data using line charts and scatterplots. While this worked fine during the early construction phase of the observatory with only a few antennas available, this solution clearly does not scale to the over a thousand baselines now formed by the almost full complement of 66 antennas. What the astronomer appreciated with Cubix was that he could clearly see patterns at once, such as antenna-based problems, without having to select and plot specific antenna pairs as he has to do now, and which is very tedious: *"With Cubix, you visualize it instantaneously. It is much faster."* He also enjoyed the fact that he could smoothly transition from one view to another. In the same vein, the neurologist commented: *"With your tool you just load the data and you see it [...] It is a very nice piece of work."* At the time of writing, both experts expressed interest in exploring their data further with Cubix. The astronomer is currently preparing additional datasets that he would like to explore with our system. The neurologist also stated that he wanted to further use Cubix in the coming weeks. The data he looked at was the result of a recent experiment, and he wanted to spend more time exploring it as to become more familiar with it.

## CONCLUSION AND FUTURE WORK

We introduced Matrix Cubes, a novel representation for dynamic weighted networks. We described how Cubix, our interactive system for the exploration of Matrix Cubes, can be used to visualize such networks by decomposing the cubes into meaningful 2D views. The design decisions made in Cubix were primarily aimed at making the cube model – and the associated interactions for manipulating it – both simple to understand and consistent across views on the data, so as to provide users with an effective metaphor.

The feedback we got from the two experts was encouraging: they both understood the cube metaphor and to navigate between views, which enabled them to make interesting observations about their data. Furthermore, the two experts stated that Cubix was the first system that actually gave them the opportunity to look at their data in an effective way. Both of them expressed interest in using it further to explore additional data in the future. However encouraging, this evaluation is only a first step and we need to further evaluate the Matrix Cube . A limitation of the study is that it only involved two domain experts. While they were from two very different domains (astronomy and neurology), Cubix should be evaluated with more experts to check that our results generalize across additional domains. Controlled experiments involving low-level network exploration and analysis tasks will also help identifying the strengths and weaknesses of the Matrix Cube approach compared to other dynamic network visualization approaches.

The Matrix Cube model is an effective metaphor that opens up a rich design space for complex data structure visualization. While the current model only supports undirected networks, we believe that it can be adapted to other data visualization problems and application domains. Possible extensions include support for multi-variate networks, as requested by one of the experts; directed networks and support for larger graphs by allowing users to interactively collapse and expand individual slices and cells.

Cubix is implemented in Java using OpenGL and is available for download at `http://www.aviz.fr/Research/Cubix`

## ACKNOWLEDGMENTS

We wish to thank the experts who used Cubix and gave us feedback: Denis Barkats from the Joint ALMA Office, National Radio Astronomy Observatory; Stephane Sockeel and

Habib Benali from the Laboratoire d'Imagerie Fonctionnelle, Université Pierre et Marie Curie Paris VI.

## REFERENCES


1. Ahn, J., Taieb-Maimon, M., Sopan, A., Plaisant, C., and Shneiderman, B. Temporal Visualization of Social Network Dynamics: Prototypes for Nation of Neighbors. In *Proc. Social Computing, Behavioral-Cultural Modeling and Prediction conference* (2011), 309–316.
2. Alper, B., Bach, B., Henry Riche, N., Isenberg, T., and Fekete, J.-D. Weighted Graph Comparison Techniques for Brain Connectivity Analysis. In *Proc. CHI*, ACM (2013), 483–492.
3. Beaudouin-Lafon, M. Designing interaction, not interfaces. In *Proc. AVI*, ACM (2004), 15–22.
4. Bezerianos, A., Chevalier, F., Dragicevic, P., Elmqvist, N., and Fekete, J. Graphdice: A system for exploring multivariate social networks. *Computer Graphics Forum 29*, 3 (2010), 863–872.
5. Boyandin, I., Bertini, E., Bak, P., and Lalanne, D. Flowstrates: An approach for visual exploration of temporal origin-destination data. *Computer Graphics Forum 30*, 3 (2011), 971–980.
6. Brandes, U., and Corman, S. R. Visual unrolling of network evolution and the analysis of dynamic discourse. *Information Visualization 2*, 1 (2003), 40–50.
7. Burch, M., Vehlow, C., Beck, F., Diehl, S., and Weiskopf, D. Parallel edge splatting for scalable dynamic graph visualization. *IEEE TVCG 17*, 12 (2011), 2344–2353.
8. Carpendale, S., Cowperthwaite, D. J., Tigges, M., Fall, A., and Fracchia, F. D. The Tardis: A Visual Exploration Environment for Landscape Dynamics. In *Proc. SPIE Conference on Visual Data Exploration and Analysis VI* (1999), 110–119.
9. Carpendale, S., Fall, A., Cowperthwaite, D., Fall, J., and Fracchia, F. Case study: visual access for landscape event based temporal data. In *Proc. Visualization*, IEEE (1996), 425–428.
10. Cassinelli, A. The Khronos Projector, June 2005. `http://www.k2.t.u-tokyo.ac.jp/members/alvaro/Khronos/`.
11. Collberg, C., Kobourov, S., Nagra, J., Pitts, J., and Wampler, K. A system for graph-based visualization of the evolution of software. In *Proc. SoftVis*, ACM (2003), 77–86.
12. Dwyer, T., and Gallagher, D. R. Visualising changes in fund manager holdings in two and a half-dimensions. *Information Visualization 3*, 4 (2004), 227–244.
13. Elmqvist, N., Dragicevic, P., and Fekete, J.-D. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *Proc. InfoVis 2008 14*, 6 (2008), 1141–1148.
14. Farrugia, M., Hurley, N., and Quigley, A. Exploring temporal ego networks using small multiples and tree-ring layouts. In *Proc. ACHI*, IARIA (2011).
15. Fels, S., Lee, E., and Mase, K. Techniques for interactive video cubism. In *Proc. Multimedia (poster session)*, ACM (2000), 368–370.
16. Friedrich, C., and Eades, P. The marey graph animation tool demo. In *Proc. Graph Drawing*, Springer-Verlag (2001), 396–406.
17. Gatalsky, P., Andrienko, N., and Andrienko, G. Interactive analysis of event data using space-time cube. In *Proc. Information Visualization*, IEEE (2004), 145–152.
18. Ghoniem, M., Fekete, J.-D., and Castagliola, P. On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization 4*, 2 (2005), 114–135.
19. Goldman, R. I., Stern, J. M., Engel, J., and Cohen, M. S. Acquiring simultaneous EEG and functional MRI. *Clinical Neurophysiology 111*, 11 (2000), 1974– 1980.
20. Gove, R., Gramsky, N., Kirby, R., Sefer, E., Sopan, A., Dunne, C., Shneiderman, B., and Taieb-Maimon, M. NetVisia: Heat Map and Matrix Visualization of Dynamic Social Network Statistics and Content. In *Proc. SocialCom/PASSAT* (2011), 19–26.
21. Hascoët, M., and Dragicevic, P. Interactive graph matching and visual comparison of graphs and clustered graphs. In *Proc. AVI*, ACM (2012), 522–529.
22. Henry, N., and Fekete, J.-D. MatrixExplorer: a Dual-Representation System to Explore Social Networks. *IEEE TVCG 12*, 5 (2006), 677–684.
23. Kapler, T., and Wright, W. GeoTime information visualization. *Information Visualization 4*, 2 (2005), 136–146.
24. Kraak, M. J. The Space-Time Cube revisited from a Geovisualization Perspective. *Proc. International Cartographic Conference* (2003), 1988–1996.
25. Liu, Z., Navathe, S., and Stasko, J. Ploceus: Network-based visual analysis of tabular data. In *Proc. VAST*, IEEE (2011), 41–50.
26. Pietriga, E., Cubaud, P., Schwarz, J., Primet, R., Schilling, M., Barkats, D., Barrios, E., and Vila Vilaro, B. Interaction Design Challenges and Solutions for ALMA Operations Monitoring and Control. In *Proc. Astronomical Telescopes and Instr.*, SPIE (2012).
27. Reda, K., Tantipathananandh, C., Johnson, A., Leigh, J., and Berger-Wolf, T. Visualizing the evolution of community structures in dynamic social networks. In *Proc. EuroVis*, Eurographics Assoc. (2011), 1061–1070.
28. Sallaberry, A., Muelder, C., and Ma, K.-L. Clustering, visualizing, and navigating for large dynamic graphs. In *Proc. Graph Drawing*, Springer (2013), 487–498.
29. Shi, L., Wang, C., and Wen, Z. Dynamic network visualization in 1.5D. In *Proc. PacificVis*, IEEE (2011), 179–186.
30. Stein, K., Wegener, R., and Schlieder, C. Pixel-oriented visualization of change in social networks. In *Proc. Int. Conf. on Advances in Social Networks Analysis and Mining*, IEEE (2010), 233–240.
31. Yi, J.-S., Elmqvist, N., and Lee, S. TimeMatrix: Visualizing Temporal Social Networks Using Interactive Matrix-Based Visualizations. *IJHCI 26*, 11-12 (2010), 1031–1051.